

# **CA314 – Object Oriented Analysis & Design - 11**

**File name:** CA314\_Section\_11\_REQ\_Ver01

**Author:** L Tuohey

**No. of pages:** 44

## Table of Contents

Explanatory Note to SRS for <i>Encounter</i> Video Game .....	6
0. Service sections.....	6
0.1 Comments and conventions re layout etc .....	6
0.2 Explanatory note .....	6
0.3 History of versions of this document .....	6
1. Introduction.....	7
1.1 Purpose.....	7
1.2 Scope.....	7
1.3 Definitions, acronyms, & abbreviations .....	7
1.4 References.....	8
1.5 Overview .....	8
2. Overall Description.....	8
2.1 Product perspective.....	9
2.1.1 Concept of operations .....	9
2.1.2 User interface concepts .....	11
2.1.2.1 <i>Area</i> user interface concept.....	11
2.1.2.2 User interface concept for setting quality values.....	11
2.1.3 Hardware interfaces .....	12
2.1.4 Software interfaces.....	12
2.1.5 Communications interfaces.....	12
2.1.6 Memory constraints .....	12
2.1.7 Operations .....	13
2.1.8 Site adaptation requirements.....	13
2.2 Product functions .....	13
2.2.1 "Initialize" use case.....	13
2.2.2 "Travel to adjacent area" use case .....	14
2.2.3 "Encounter foreign character" use case .....	15
2.3 User characteristics .....	15
2.4 Constraints .....	15
2.5 Assumptions and dependencies .....	15
2.6 Apportioning of requirements.....	15
3. Specific Requirements .....	16
3.1 External interface requirements .....	16
3.1.1 User interfaces .....	16
3.1.2 Hardware interfaces .....	18
3.1.3 Software interfaces.....	18
3.1.4 Communication interfaces .....	18
3.2 Specific requirements.....	18
3.2.1 Sequence diagrams.....	18
3.2.1.1 <i>Initialize</i> use case .....	18
3.2.1.2 Travel to adjacent area use case.....	19
3.2.1.3 Engage foreign character use case .....	20
3.2.2 Classes for classification of specific requirements .....	21
3.2.AR <i>Areas</i> .....	22
3.2.AR.1 Attributes of areas .....	23
3.2. AR.1.1 <i>Area name</i> .....	23
3.2.AR.1.2 <i>Area image</i> .....	23
3.2.AR.1.3 <i>Area-specific qualities</i> .....	23
3.2.AR.2 Area entities.....	23

3.2.AR.2.1	<i>Courtyard area</i>	24
3.2.AR.2.2	<i>Dressing room area</i>	24
3.2.AR.2.3	<i>Dungeon area</i>	25
3.2.AR.2.4	<i>Kitchen area</i>	26
3.2.AR.2.5	<i>Living room area</i>	27
3.2.AR.2.6	<i>Study area</i>	28
3.2.AR.3	Area functionality	29
3.2.AR.4	Events pertaining to areas	29
3.2.AR.4.1	<i>Display on entry of player character</i>	29
3.2.AR.4.2	<i>Handling engagements</i>	29
3.2.AR.4.7	<i>Display on entry of foreign character</i>	30
3.2.CH	Connection hyperlinks between areas	30
3.2.CH.1	Attributes of connections hyperlinks	30
3.2.CH.1.1	<i>Connection</i>	30
3.2.CH.2	Connection hyperlink entities	30
3.2.CH.3	Functionality of connection hyperlinks	30
3.2.CH.4	Events pertaining to connection hyperlinks	30
3.2.CH.4.1	<i>User clicks on a connection hyperlink</i>	30
3.2.CO	Connections between areas	30
3.2.CO.1	Attributes of connections between areas	31
3.2.CO.1.1	<i>First and second areas</i>	31
3.2.CO.2	Connections entities	31
3.2.CO.2.1	<i>Dressing room - courtyard</i>	31
3.2.CO.2.2	<i>Dungeon - study</i>	31
3.2.CO.2.3	<i>Study - living room</i>	31
3.2.CO.2.4	<i>Courtyard - living room</i>	32
3.2.CO.2.5	<i>Dressing room - dungeon</i>	32
3.2.CO.2.6	<i>Courtyard - kitchen</i>	32
3.2.CO.3	Functionality of area connections	32
3.2.CO.4	Events pertaining to area connections	32
3.2.CO.4.1	<i>Moving a character through a connection</i>	32
3.2.EC	Encounter characters	32
3.2.EC.1	Attributes of Encounter characters	32
3.2.EC.1.1	<i>Name of Encounter characters</i>	32
3.2.EC.1.2	<i>Qualities of Encounter characters</i>	32
3.2.EC.1.3	<i>Image of Encounter characters</i>	33
3.2.EC.2	Encounter character entities	33
3.2.EC.3	Functionality of <i>Encounter</i> characters	33
3.2.EC.3.1	<i>Living points</i>	33
3.2.EC.3.2	<i>Configurability of Encounter character quality values</i>	33
3.2.ED	Engagement displays	33
3.2.ED.4	Engagement display events	34
3.2.ED.4.1	<i>Dismissing the display</i>	34
3.2.EG	The <i>Encounter</i> game	34
3.2.EG.1	Attributes of the <i>Encounter</i> game	34
3.2.EG.1.1	<i>Duration</i>	34
3.2.EG.1.2	<i>Action buttons</i>	34
3.2.EG.2	Entities of the <i>Encounter</i> game	35
3.2.EG.2.1	<i>Single game</i>	35
3.2.EG.4	Events the <i>Encounter</i> game	35

3.2.EG.4.4 Pressing the Set qualities button .....	35
3.2.EG.4.5 Pressing the End game button .....	35
3.2.EG.4.6 Pressing the Get status button .....	35
3.2.EN Engagements.....	35
3.2.EN.1 Attributes of engagements .....	35
3.2.EN.2 Engagement entities.....	35
3.2.EN.3 Functionality of engagements.....	36
3.2.EN.3.1 Engaging a foreign character.....	36
3.2.EN.4 Events on engagements .....	36
3.2.EN.4.1 Interrupting engagements .....	36
3.2.FC Foreign characters .....	37
3.2.FC.1 Attributes of foreign characters.....	37
3.2.FC.2 Foreign character entities .....	37
3.2.FC.2.1 Freddie foreign character.....	37
3.2.FC.3 Functionality of foreign characters .....	38
3.2.FC.3.1 Foreign character movement.....	38
3.2.PC Player characters .....	38
3.2.PC.1 Attributes of player characters .....	38
3.2.PC.2 Player character entities .....	39
3.2.PC.2.1 Player's main character.....	39
3.2.PC.2.2 Additional characters under the control of the player .....	39
3.2.PC.3 Player character functionality .....	39
3.2.PC.3.1 Configurability of the player character quality values.....	39
3.2.PC.3.2 Configurability of the player character images.....	39
3.2.PC.3.3 Aging of the player character images.....	39
3.2.PQ The player quality window .....	39
3.2.PQ.1 Attributes of the player quality window .....	40
3.2.PQ.2 Player quality window entity .....	40
3.2.PQ.2.1 Window for allocating qualities .....	40
3.2.PQ.3 Player quality functionality.....	41
3.2.PQ.3.1 Initiating the display.....	41
3.2.PQ.4 Player quality window events.....	41
3.2.PQ.4.1 Displaying the value of a quality .....	41
3.2.PQ.4.2 Setting the value of a quality .....	41
3.2.PQ.4.3 Dismissing the window .....	41
3.2.PQ.4.4 Interruption.....	41
3.2.TH Thumbnail sketch of playing areas.....	41
3.2.TH.1 Attributes of the thumbnail sketch.....	41
3.2.TH.1.1 Thumbnail image .....	41
3.3 Performance requirements .....	42
3.4 Design constraints .....	42
3.5 Software system attributes .....	43
3.5.1 Reliability.....	43
3.5.2 Availability .....	43
3.5.3 Security .....	43
3.5.4 Maintainability.....	43
3.5.4.1 Changing characters and areas.....	43
3.5.4.2 Globally altering styles .....	43
3.5.4.3 Altering rules of engagement.....	43
3.6 Other requirements.....	43

4. Supporting information.....	43
4.1 Table of contents and index .....	43
4.2 Appendixes .....	43

## **Explanatory Note to SRS for *Encounter* Video Game**

This case study is taken from **Reference 3** “Software Design, from Programming to Architecture”, Braude (Wiley).

The case study is presented in two documents:

Software Requirements Specification (**SRS**) [*present document*]

Software Design Document (**SDD**)

Both the SRS and SDD may be thought of as being made up of two parts. In the case of the SRS we have

**Part 1** (sections 1, 2): **Customer (C) Requirements**

**Part 2** (sections 3, 4): **Specific (D) Requirements**

Note: The section numbering of Reference 1 is retained, that is, a prefix “11” has not been prepended to the section numbers.

### **0. Service sections**

#### **0.1 Comments and conventions re layout etc**

[Note: Using a standard to write the SRS (software requirement specification) helps one to cover all of the aspects of requirements that readers need to know about, and provides a recognized structure. Several standards are available, but we will concentrate on the IEEE standard (see IEEE 830-1993 if interested). Most organizations allow modification of the standards to tailor it for their own use. The template used below modifies the standard by omitting some less important sections and by adding sections on concept of operations and use cases.]

#### **0.2 Explanatory note**

[*Note to the student*: Sections 1 and 2 cover the customer (C-) requirements. The remainder of the document, sections 3 and 4, contain the specific (D-) requirements. The convention used here is that C-requirements are not intended to be detailed enough to develop the design and implementation: this is the purpose of the D-requirements.]

#### **0.3 History of versions of this document**

Omitted

# 1. Introduction

## 1.1 Purpose

[*Note to the student:* the purpose of *this* entire document (not the purpose of the application).]

This document provides all of the requirements for the *Encounter* video game. Sections one and two are intended primarily for customers of the application, but will also be of interest to software engineers building or maintaining the software. Section three is intended primarily for software engineers, but will also be of interest to customers.

## 1.2 Scope

[*Note to the student:* what overall aspects of the application this document is intended to cover.]

This document covers the requirements for release 1.0 of *Encounter*. Mention will be made throughout this document of selected probable features of future releases. The purpose of this is to guide developers in selecting a design that will be able to accommodate the full-scale application.

## 1.3 Definitions, acronyms, & abbreviations

Acronym or term	Definition
Alive	A game character is said to be "alive" if it has at least one quality with non-zero value.
C-requirement	Statement of the requirements for the application, expressed in a form clear to the customer.
D-requirement	Statement of the requirements for the application, given in a form detailed enough to be used by the developers for design and implementation. If possible, D-requirements should also be understandable to the customer.
<i>Encounter</i>	Name of this application; also, a meeting between two game characters in an area (but not necessarily an "engagement" -- see below)
Engagement	An interaction between characters of the game, which typically affect the characters
RPG	"Role-playing game"; a game, typically played on a computer, in which the players adopt character roles
Role-playing game	See RPG
Video game	A game played on a computer

Table 3.3 Glossary of terms, Acronyms

## 1.4 References

[Note to student: Of the following, we will look at SDD only; for those interested refer to book website]

Software Configuration Management Plan (SCMP) for *Encounter version 1.0*

Software Design Description (SDD) for *Encounter version 1.0*

Software Project Management Plan (SPMP) for *Encounter version 1.2*

Software Quality Assurance Plan (SQAP) for *Encounter version 1.0*

Software User Documentation Plan (SUDP) for *Encounter version 1.0*

Software Test Documentation (STD) for *Encounter version 1.0*

## 1.5 Overview

Intentionally omitted.

[Note to the student: the author of this document felt no need for this section, intending to cover its contents in section 2.]

## 2. Overall Description

[*Note to the student:* make this general enough that it is unlikely to change much in future versions. Avoid statements that are repeated in later sections.]

*Encounter* is to be a role-playing game which simulates all or part of the lifetime of the player's main character. It should be of interest to both men and women. The measure of "success" in playing *Encounter* is up to the player. Typically, success will be measured by the "life points" maximum attained by the player or by the ability of the player to live as long a life as possible.

Some game characters are to be under the control of the player. The rest, called "foreign" characters, are to be under the application's control. Game characters will have a fixed total number of points allocated among qualities such as *strength*, *stamina*, *patience* etc. Characters encounter each other when they are in the same area at the same time, and may then engage each other. The result of the engagement depends on the values of their qualities and on the environment in which the engagement takes place. Engagements are not necessarily violent or adversarial.

Players have restricted opportunities to reallocate their qualities. One of the player-controlled characters will be referred to as the "main" player character.

In early versions of this game, there will be only one player-controlled character, and one foreign character.

The eventual nature of the characters is to be determined from insights gained from surveys and focus groups. It is expected that initial releases will not have animation.

*Encounter* should eventually be highly customizable, so that users can either start with predefined games, substitute pre-designed characters and rules of engagement, or devise their own characters and rules of engagement.

The design should support expansion into a family of games, including Internet-based multiple player versions.

## 2.1 Product perspective

[*Note to the student:* In this section, *Encounter* is compared with other related or competing products. This is a useful way to provide perspective on the application. Subheading 2.1.1. of this section has been changed from the IEEE standard to accommodate "concept of operations".]

*Encounter* is intended to fulfill the need for gamers to have a greater influence over the contents of video games with and without programming. It is also intended for a somewhat mature clientele. *Encounter* is intended to appeal to both genders. The design and documentation for *Encounter* will make it convenient to expand and modify the game. It is anticipated that *Encounter* will be used as a legacy application for expansion into applications such as office interaction simulations.

### 2.1.1 Concept of operations

[*Note to the student:* This section conveys the overall concept of the application by whatever means are most natural for doing so. In the case of *Encounter*, the requirements developers decided that state/transitions best convey the concept.]

*Encounter* can be in one of several states, as shown in [figure 3.40](#).

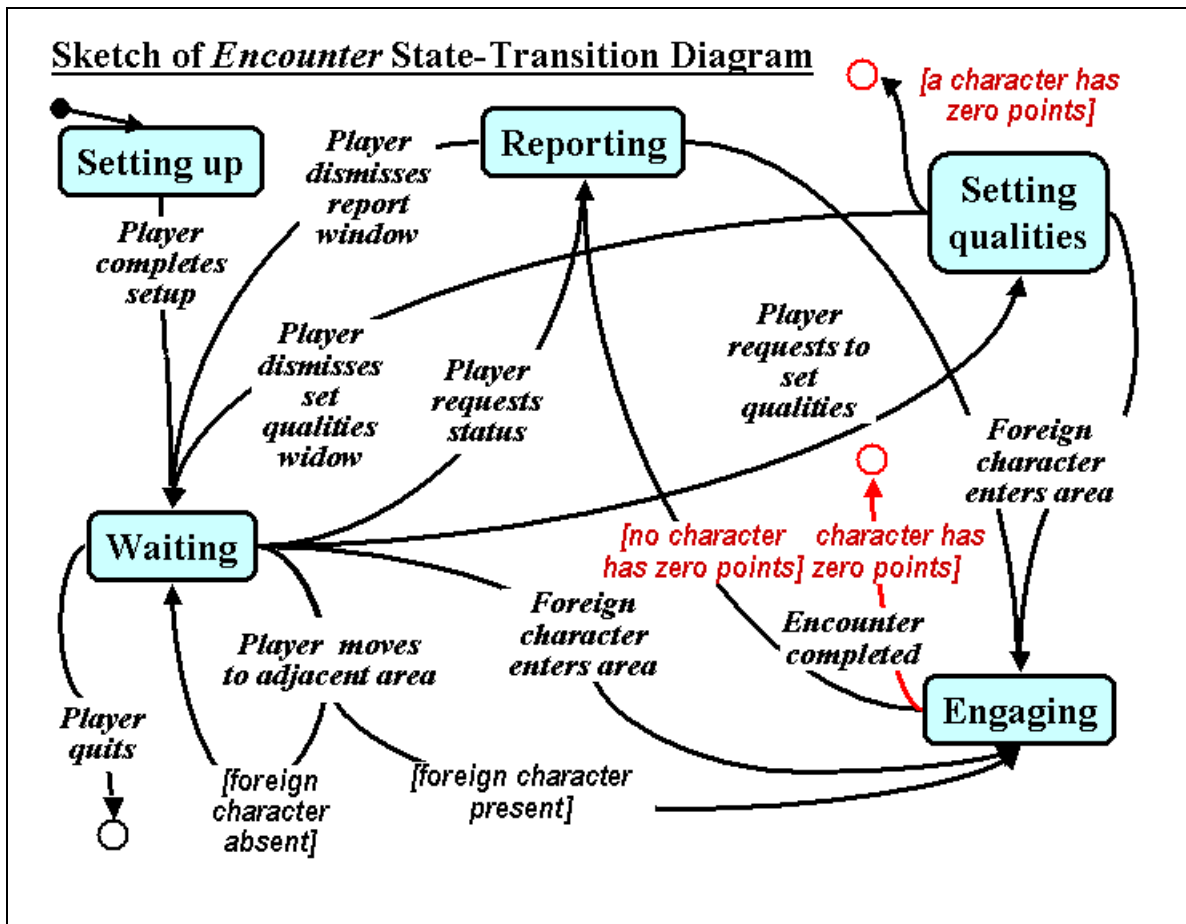


Figure 3.40 *Encounter* State-Transition Diagram

- *Setting up*: the state in which the game is being set up by the player
- *Reporting*: the system is displaying a window showing the status of the player's character(s)
- *Setting qualities*: equipping the player's character with qualities. This process consumes mandatory amounts of time, and can be performed as long as no foreign character is present.
- *Engaging*: the state which applies whenever a foreign character and the player's main character are both present in an area at the same time
- *Waiting*: The player and the foreign character(s) are not active

This state/transition is tested by integration test <test reference goes here>.

## 2.1.2 User interface concepts

[*Note to the student:* the following figures are preliminary sketches of key user interfaces only, used to provide perspective on the product. All the user interfaces are specified in detail in section 3. We have modified the actual IEEE heading, "user interfaces", to emphasize that these are not the detailed UI's.]

### 2.1.2.1 Area user interface concept

The areas in which encounters take place shall have an appearance very roughly like that shown in [figure 3.41](#).

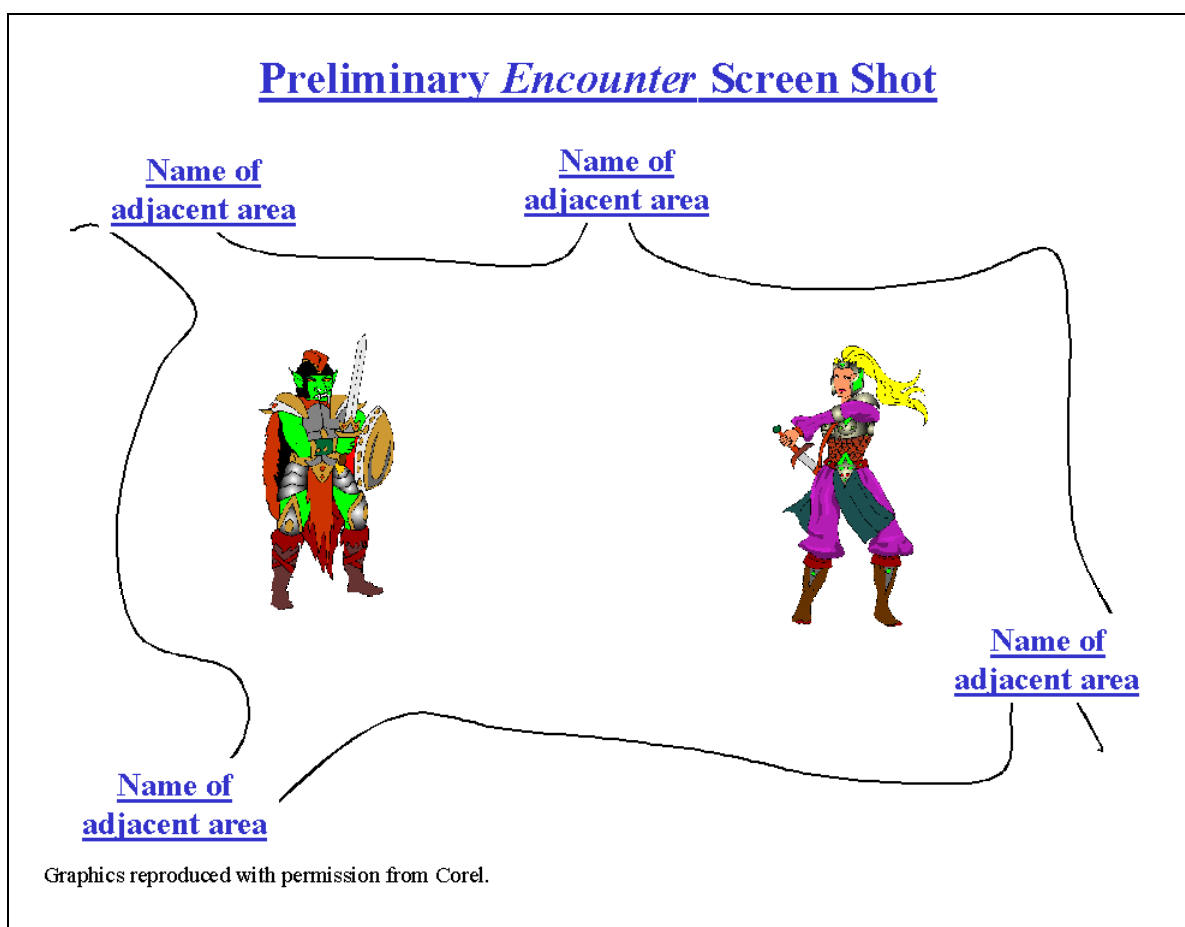


Figure 3.41 Preliminary *Encounter* Screen Shot

### 2.1.2.2 User interface concept for setting quality values

When setting the values of game characters under his control, the player will retrieve an interface of the form sketched approximately in [figure 3.42](#). The scroll box is used to identify the quality to be set, and the text box is used for setting the value.

## Preliminary Sketch of User Interface for Setting Game Character Qualities

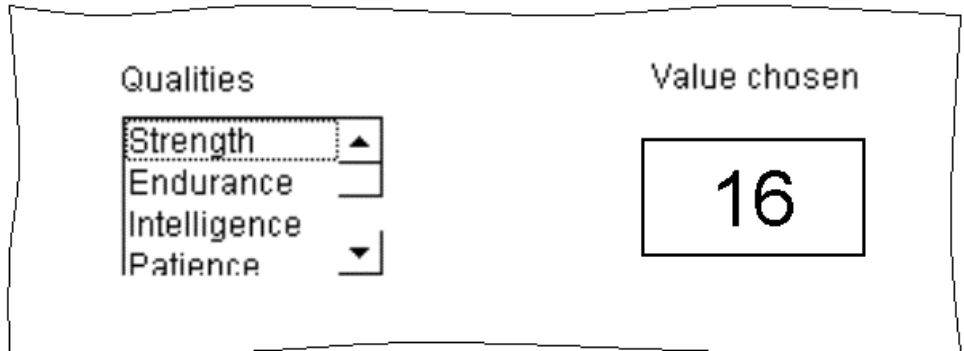


Figure 3.42 Preliminary Sketch of User Interface for Setting Game Character Qualities

### **2.1.3 Hardware interfaces**

Future releases will utilize a joystick.

### **2.1.4 Software interfaces**

Java virtual machine capable of executing Java 1.1 or higher.

### **2.1.5 Communications interfaces**

None. Future releases will interface with the Internet via a modem.

### **2.1.6 Memory constraints**

*Encounter* shall require no more than 16 MB of RAM and 20MB of secondary storage. (see test plan <test reference>).

## 2.1.7 Operations

[*Note to the student:* Normal and special operations required by the user]

[Future release] It shall be possible to save and retrieve a game.

## 2.1.8 Site adaptation requirements

[*Note to the student:* Requirements for execution on a particular installation; versions in various languages (e.g., French, Japanese, Spanish) etc.]

None

## 2.2 Product functions

[*Note to the student:* Summary of the major functions of the application. More detailed than section 1.5; less detailed than section 3. The writers of this SRS decided that use cases are an appropriate manner in which to specify the major overall functionality.]

This section specifies the required overall functionality of the application, but is not intended to provide the complete specifications. Section 3 provides the requirements in complete detail.

### 2.2.1 "Initialize" use case

Actor: player of *Encounter*

Use case: [figure 3.43](#) gives the text of the *Initialize* use case. The use case is shown in context with the *Encounter foreign character* use case and the *Set rules* use cases. *Initialize* is the typical sequence users execute at the beginning of a session.

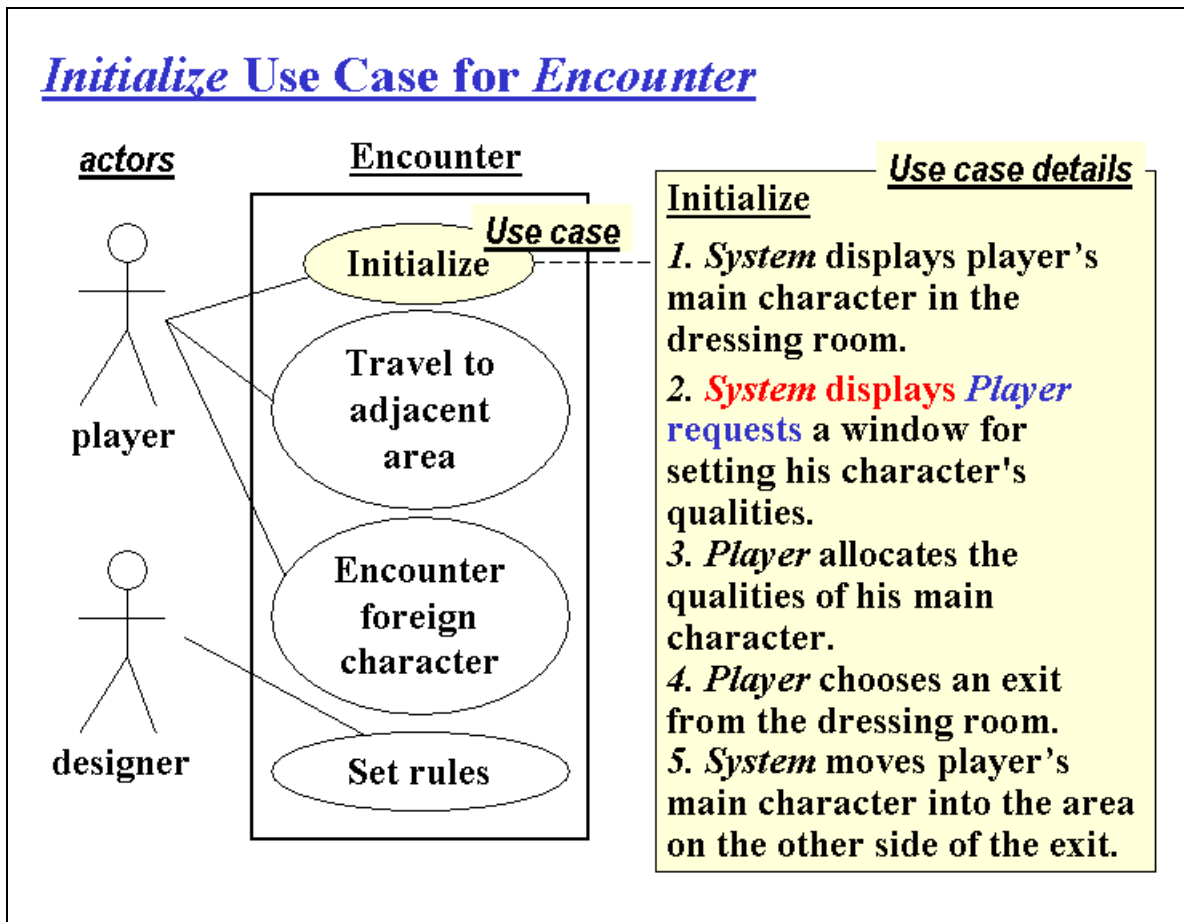


Figure 3.43 *Initialize Use Case for Encounter*

This use case corresponds to test <test reference> in the Software Test Documentation.

### 2.2.2 "Travel to adjacent area" use case

Actor: player of *Encounter*

Use case:

1. *Player* hits hyperlink connecting displayed area to adjacent area
2. *System* displays the indicated adjacent area containing player's character

### 2.2.3 "Encounter foreign character" use case

Actor: player of *Encounter*

Use case:

1. *System* moves a foreign game character into the area occupied by the player.

**Or**

*Player* moves into an area containing a foreign character.

2. *System* causes the two characters to engage
3. *System* displays the result of the engagement
4. If either the player's character or the foreign character has no points, the game terminates, otherwise
5. *System* moves the player's character to a random area different from that in which the encounter took place, and displays it there.

## 2.3 User characteristics

[*Note to the student:* indicate what kind of person the typical users are likely to be. Examples: novice, software professional, accountant with 5 years of computer usage, etc.]

The user is expected to be approximately 20-35 years of age.

## 2.4 Constraints

[*Note to the student:* all conditions that may limit the developer's options. These can originate from many sources.]

*Encounter* shall operate on PC's running Windows 95 or later at a minimum speed of 100MHz. Java shall be the implementation language.

## 2.5 Assumptions and dependencies

[*Note to the student:* any assumptions being made e.g. future hardware]

None

## 2.6 Apportioning of requirements

[*Note to the student:* Order in which requirements are to be implemented.]

The requirements described in sections 1 and 2 of this document, are referred to as "C-requirements"; those in section 3 are referred to as "D-requirements". The primary audience for C-requirements is the customer community, and the secondary audience

is the developer community. The reverse is true for the D-requirements. These two levels of requirements are intended to be consistent. Inconsistencies are to be logged as defects. In the event a requirement is stated within both the C-requirements and the D-requirements, the application shall be built from the D-requirement version, since it is more detailed.

"Essential" requirements (referred to in section 3) are to be implemented for this version of *Encounter*. "Desirable" requirements are to be implemented in this release if possible, but are not committed to by the developers. It is anticipated that they will be part of a future release. "Optional" requirements will be implemented at the discretion of the developers.

### **3. Specific Requirements**

#### **3.1 External interface requirements**

##### **3.1.1 User interfaces**

*[Note to the student:*

Section 2.1.2 in the SRS for the *Encounter* video game, showed only sketches of user interfaces in order to provide product perspective. It was not the last word, and lacked details.

If user interfaces are not completely specified later in this document, then all details should be given in this section. Since we are using the object style of specification in this case study, the details of each window are packaged with their classes in section 3.2.2.

In any case, this section should explain the physical relationships among graphical elements (e.g., cascading, tiled, superimposed). ]

*Encounter* takes place in areas. [Figure 4.54](#) shows a typical screen shot of the *courtyard* area, with a player-controlled character (?), and the results of an engagement (?):

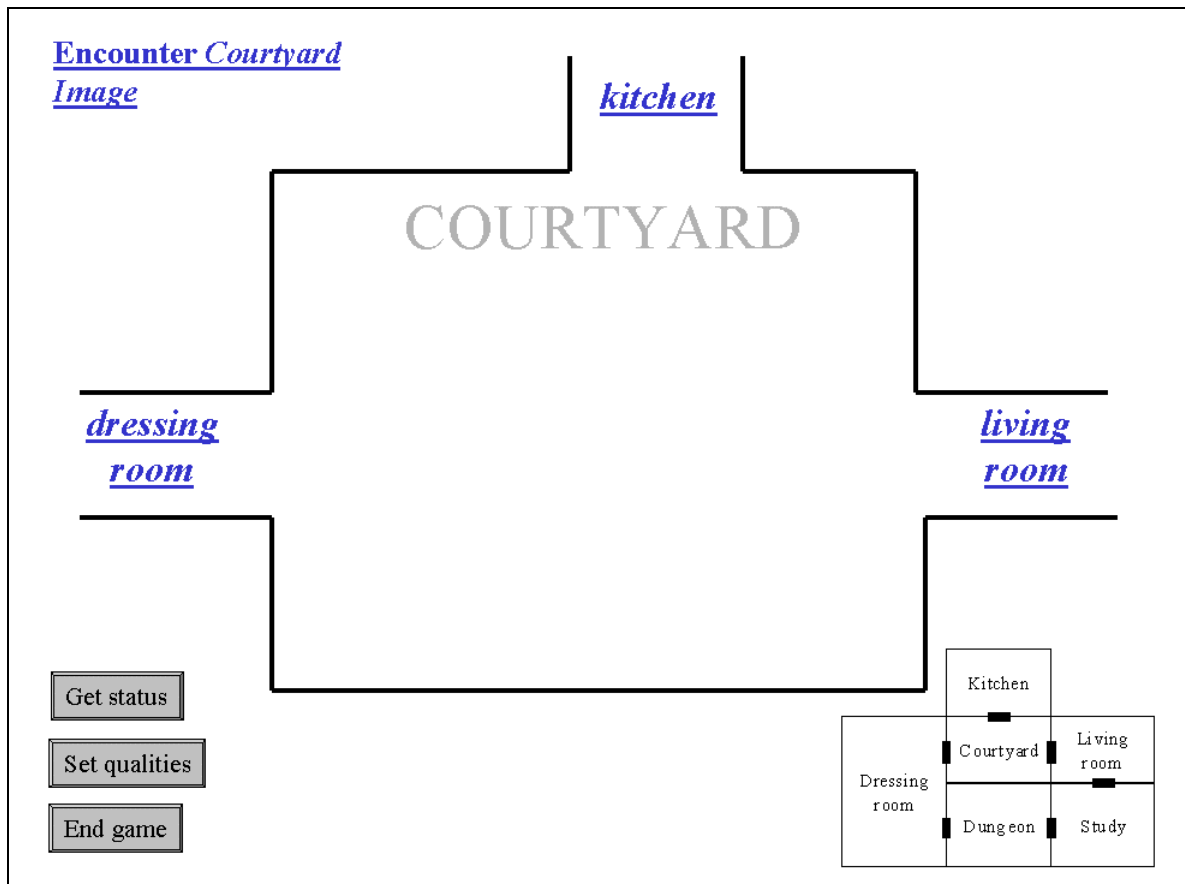


Figure 4.54 *Encounter Courtyard* Image, Including Game Characters and Status Window

Note that each part of this figure is specified separately in section 3. This interface takes up the entire window. Areas have connections to adjacent areas, labeled by hyperlinks. Clicking on one of these hyperlinks moves the player's character into the corresponding area.

The entire set of interfaces is as follows:

- a. One image for each area, specified in section 3.2AR below.
- b. A user interface for setting the quality values of the player's character, specified in section 3.2.PQ.
- c. A user interface for displaying the results of an engagement, specified in section 3.2.ED. The same user interface is used to show the status of the player's character.
- d. A window announcing the start of an engagement.

An interface of type *a* above will always be present on the monitor. When called for by these requirements, interfaces of type *b* or *c* will be superimposed.

This requirement is tested in Software Test Documentation (STD) < reference to test goes here >.

### 3.1.2 Hardware interfaces

[*Note to the student:* the hardware with which *Encounter* (which is a software application) must deal.]

none

[Future release: *Encounter* shall be controllable by a joystick.]

### 3.1.3 Software interfaces

[*Note to the student:* Other software with which *Encounter* must interface: an example would be a printer driver.]

Java virtual machine for Java 1.1 or higher.

[Future release: *Encounter* shall be playable from Intergalactic Internet Gaming Site]

### 3.1.4 Communication interfaces

none

[Future release: *Encounter* shall interface with the Internet via a modem with at least 56 Kb/s]

## 3.2 Specific requirements

[*Note to the student:* This section takes some liberties with the IEEE standards in order to account for use cases. First it describes the sequence diagrams required to express the use cases of section 2.2 of this SRS (section 3.2.1). The classes required to express these use cases are then used to classify the detailed requirements (section 3.2.2)

### 3.2.1 Sequence diagrams

[*Note to the student:* here we show a sequence diagram for each of the use cases identified in section 2.1 of the SRS. The IEEE SRS standard does not have a section called "sequence diagrams" it has been tailored to accommodate this concept.]

#### 3.2.1.1 Initialize use case

The sequence diagram for the *Initialize* use case is shown in [figure 4.55](#).

### Conceptual Sequence Diagram for *Initialize* Use Case

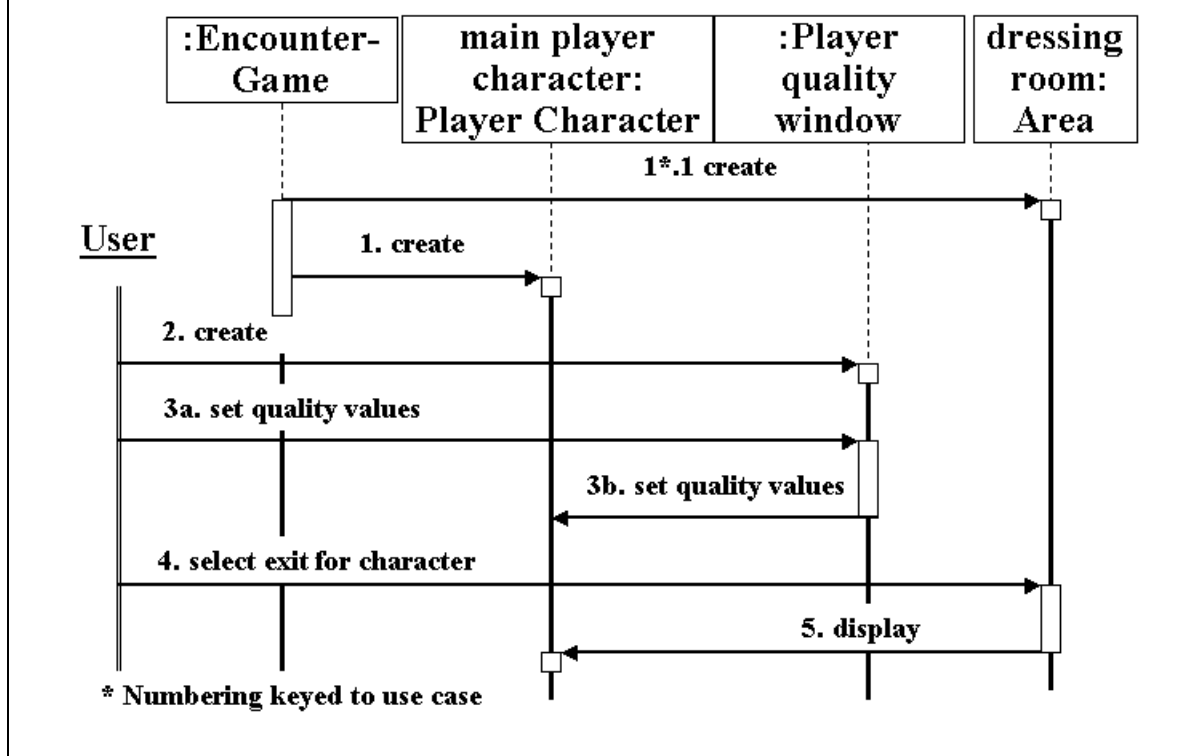


Figure 4.55 Sequence Diagram for *Initialize* Use Case

This use case requires classes *EncounterGame* (of which there is only one instance), *PlayerCharacter* (with instance main player character), *PlayerQualityWindow* (of which there is only one instance), and *Area* (with instance dressing room).

#### **3.2.1.2 Travel to adjacent area use case**

The sequence diagram for the *Travel to adjacent area* use case is shown in [figure 4.56](#).

## Conceptual Sequence Diagram for Travel to Adjacent Area Use Case

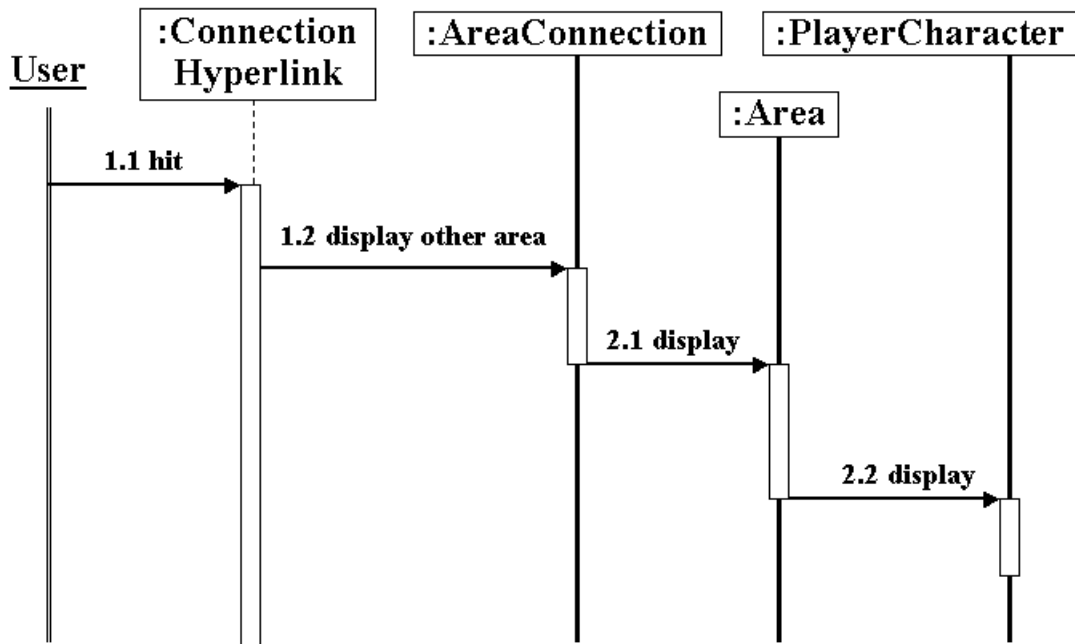


Figure 4.56 Sequence Diagram for *Tavel to Adjacent Area* Use Case

This use case requires classes *ConnectionHyperlink*, *AreaConnection*, *Area*, and *PlayerCharacter*.

### 3.2.1.3 Engage foreign character use case

The sequence diagram for the *Engage foreign character* use case is shown in the [figure 4.57](#).

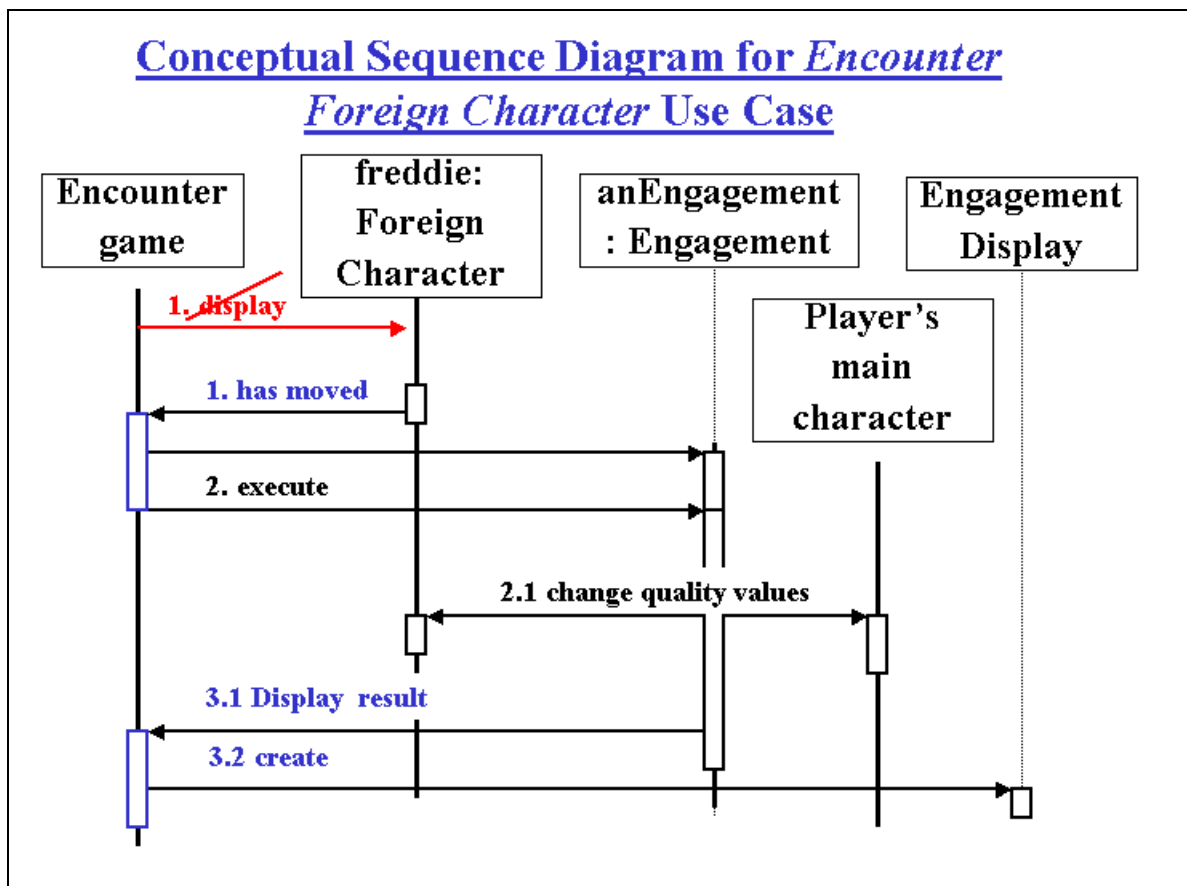


Figure 4.57 Sequence Diagram for *Encounter Foreign Character* Use Case

This use case requires classes *EncounterGame* (of which there is only one instance), *ForeignCharacter* (with instance *freddie*), *Engagement*, *PlayerCharacter*, *PlayerQualityWindow*, and *EngagementDisplay*.

### 3.2.2 Classes for classification of specific requirements

[*Note to the student:* since we are classifying the detailed requirements by class, we first list the classes that we have (very carefully!) chosen. These are not the total set of classes that will be used by the application -- merely the core classes pertaining to the domain of the application, which are adequate for organizing all of the requirements. In this case, for example, all of them are *Encounter* video game terms.]

The classes for the *Encounter* video game sufficient for expressing the requirements are *Area*, *EncounterCharacter*, *EncounterGame*, *Engagement*, *EngagementDisplay*, *ForeignCharacter*, *PlayerCharacter*, *PlayerQualityWindow*, *ConnectionHyperlink*, *AreaConnection*, and *ThumbnailMap*. These are shown in the object model of [figure 4.58](#).

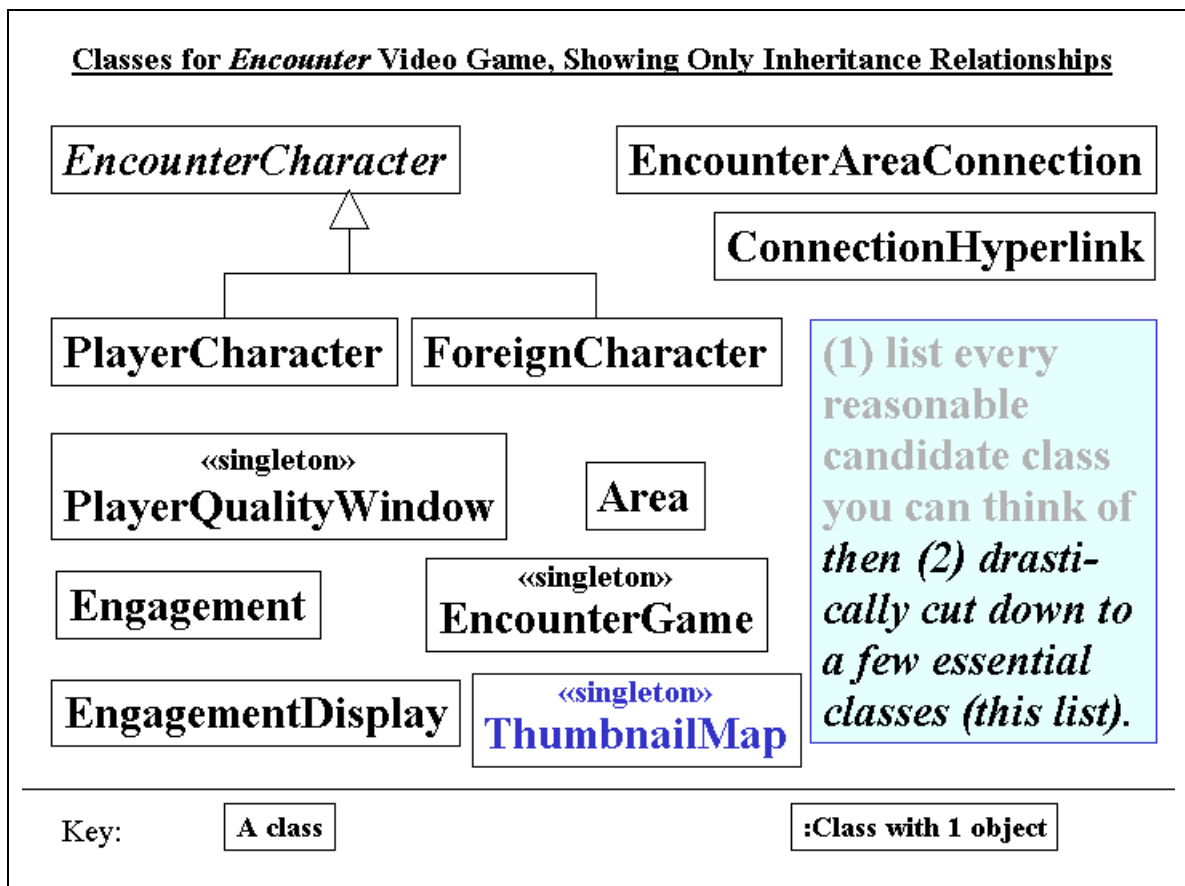


Figure 4.58 Classes for *Encounter* Video Game, Showing Only Inheritance Relationship

[*Note to the student:* The numbering used "3.2.Area.N.N..." etc. used in section 3.2 makes it easier for us to insert, remove, and locate requirements by organizing alphabetically the classes that contain them. Think in terms of hundreds of requirements. If we were to number the classes using "3.2.1....", "3.2.2..." etc., then inserting new classes would have to be done at the end of the list, since existing numbering, already referred to elsewhere in the project, could not be disturbed. The requirements would not be alphabetically ordered. As a result, one would have to go through the requirements one by one to locate a particular one.]

### 3.2.AR Areas

[*Note to the student:* first, we describe what the class (i.e., this classification of requirements) refers to.]

An area is a place viewable on the monitor. All activities of *Encounter* (including engagements) take place in areas. Rooms, gardens and courtyards are examples of areas.

### **3.2.AR.1 Attributes of areas**

[*Note to the student:* here we tell what properties each object (specific entity) of the class must possess.]

#### **3.2. AR.1.1 Area name**

[essential; implemented]

[Notes to the student:

The bracketed statement above indicates the priority and the status of the requirement. Once the requirement is coded and tested, the statement "not yet implemented" is either deleted or changed to "implemented".

"Essential" requirements are implemented first. Once a requirement has been designed for and implemented, "essential" can be removed. This is one technique for tracking the state of the application and its relationship with this SRS. Another option is to specify the iteration to which the requirement applies.]

Every area shall have a unique [case-insensitive](#) name consisting of 1 to 15 characters. Acceptable characters shall consist of [any digits and characters as specified by the ISO Unicode standard](#).

Test plan < reference to test goes here>.

[*Notes to the student:*

Each attribute-type requirement maps to a pair of *get-* and *set-* functions.

This document suggests how each requirement can be hyperlinked to a unit test in the Software Test Documentation.]

#### **3.2.AR.1.2 Area image**

[essential; implemented] There shall be an image to display each *Area* object on the part of the window not occupied by the required buttons or the thumbnail map.

#### **3.2.AR.1.3 Area-specific qualities**

[essential; implemented] Only some game character qualities shall be applicable in each area. The specific qualities required for each area are specified in section [3.2.AR.2](#).

### **3.2.AR.2 Area entities**

[*Note to the student:* we specify specific area objects that must exist within the application.]

### 3.2.AR.2.1 Courtyard area

[essential; implemented] There shall be an *Area* object with name "courtyard", requiring qualities *stamina*, and *strength*. The courtyard image is shown in [figure 4.59](#).

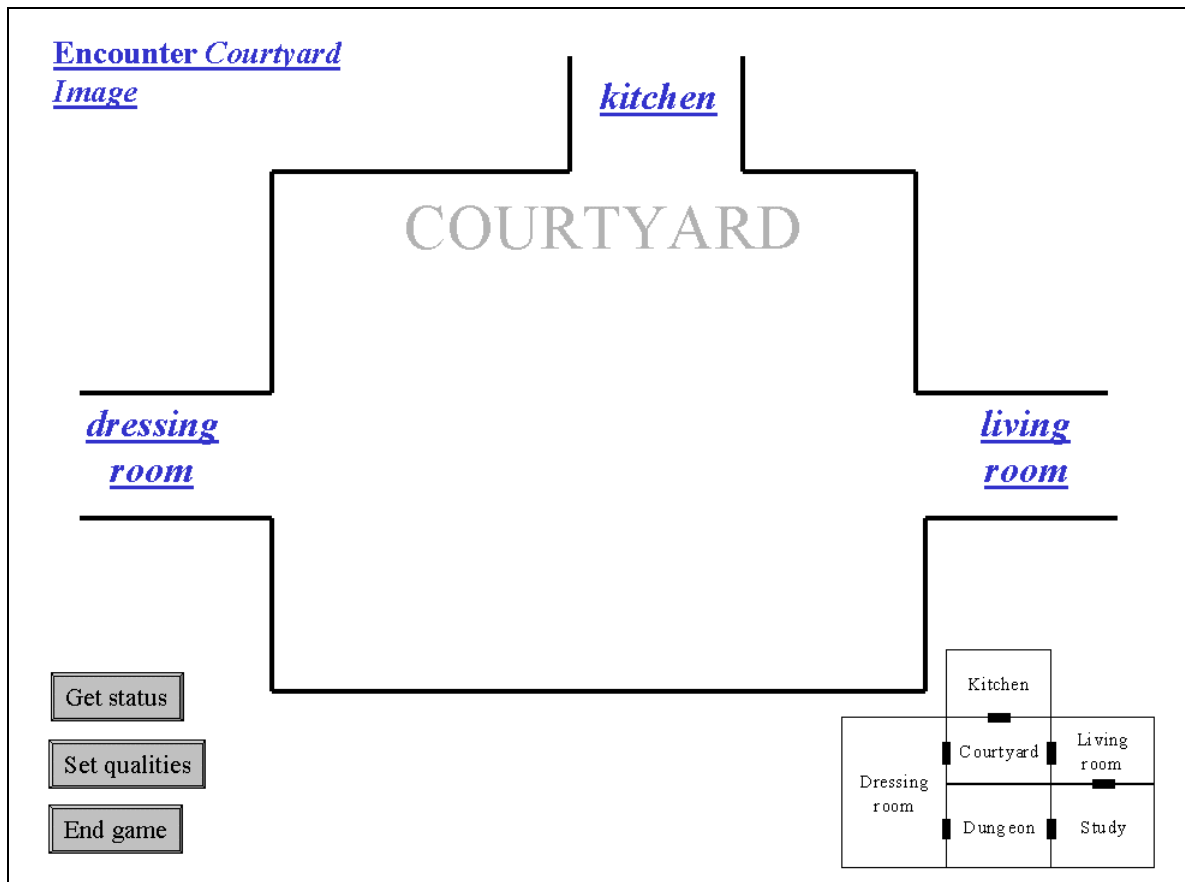


Figure 4.59 Encounter Courtyard Image

### 3.2.AR.2.2 Dressing room area

[essential; implemented] There shall be an area with name "dressing room", requiring no qualities. The image is shown in the [figure 4.60](#).

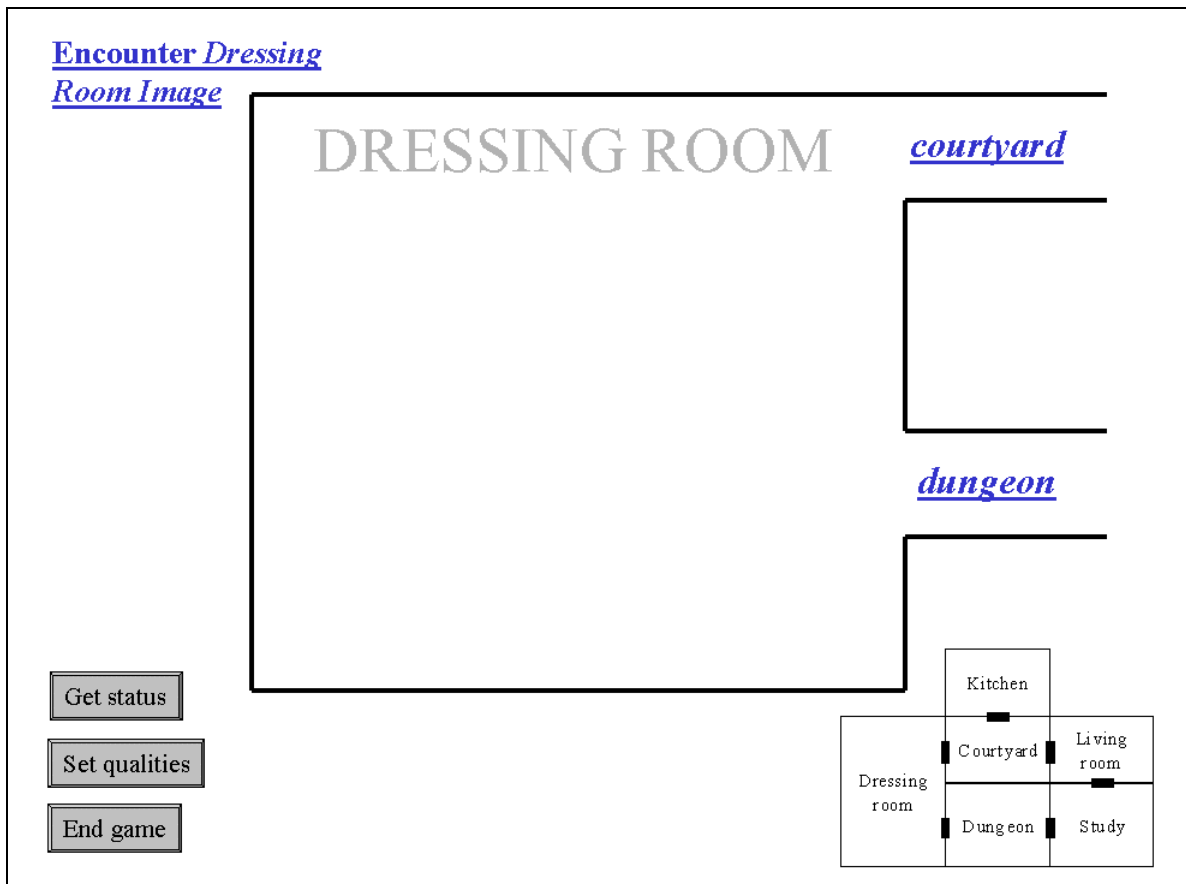


Figure 4.60 Encounter Dressing Room Image

### 3.2.AR.2.3 Dungeon area

[essential; implemented] There shall be an area with name "dungeon", requiring qualities *stamina*, and *patience*. Its image is shown in [figure 4.61](#),

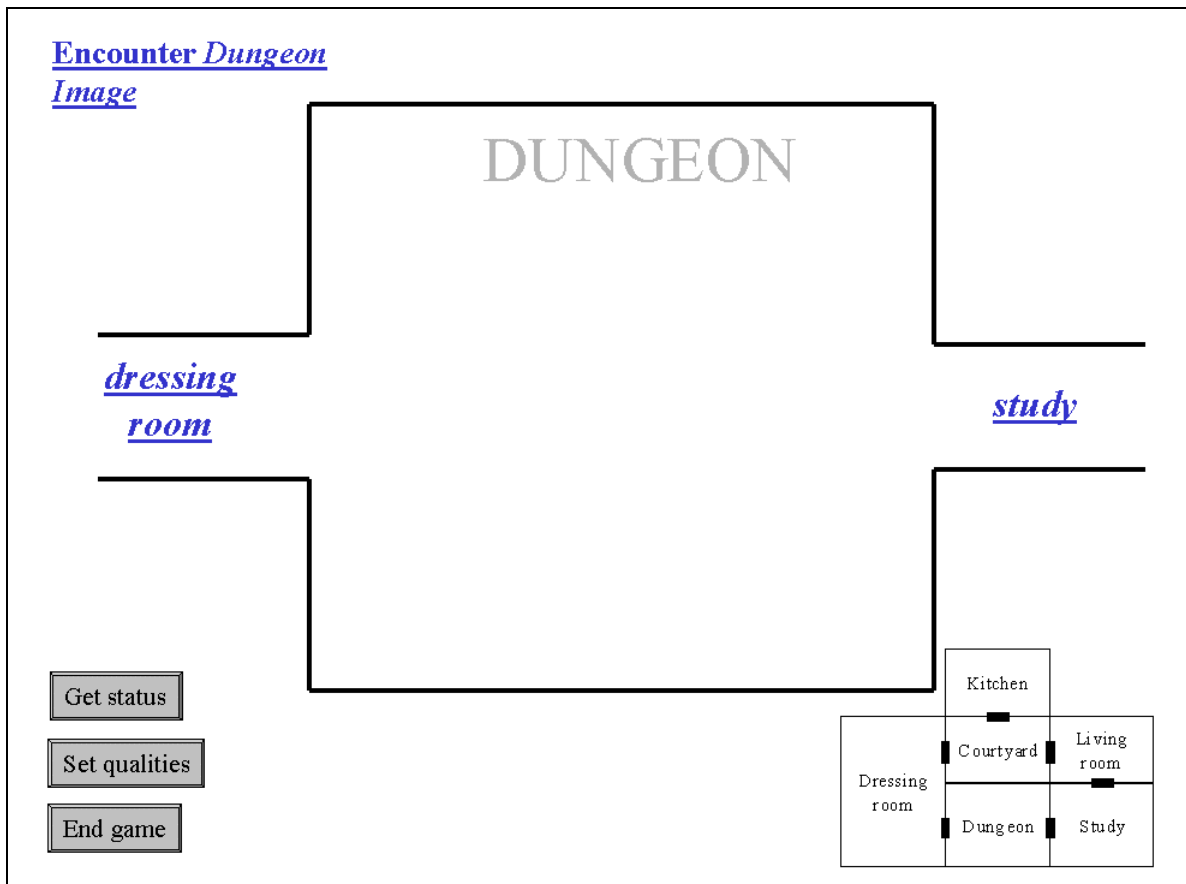


Figure 4.61 *Encounter Dungeon Image*

### **3.2.AR.2.4 Kitchen area**

[essential; implemented] There shall be an area with name "kitchen", requiring the quality *concentration*. The kitchen image is shown in [figure 4.62](#),

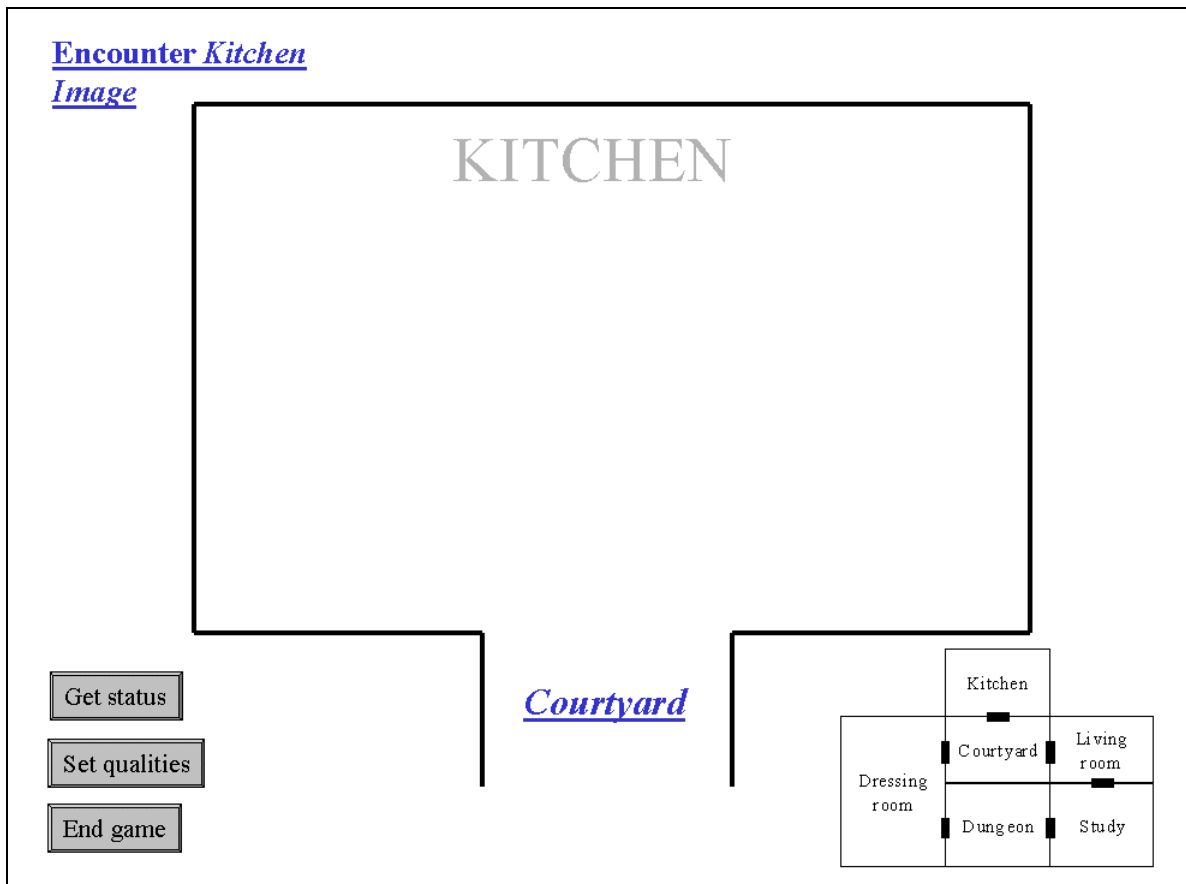


Figure 4.62 *Encounter Kitchen* Image

### 3.2.AR.2.5 *Living room area*

[essential; implemented] There shall be an area with name "living room", requiring qualities *concentration*, and *stamina*. Its image is shown in [figure 4.63](#).

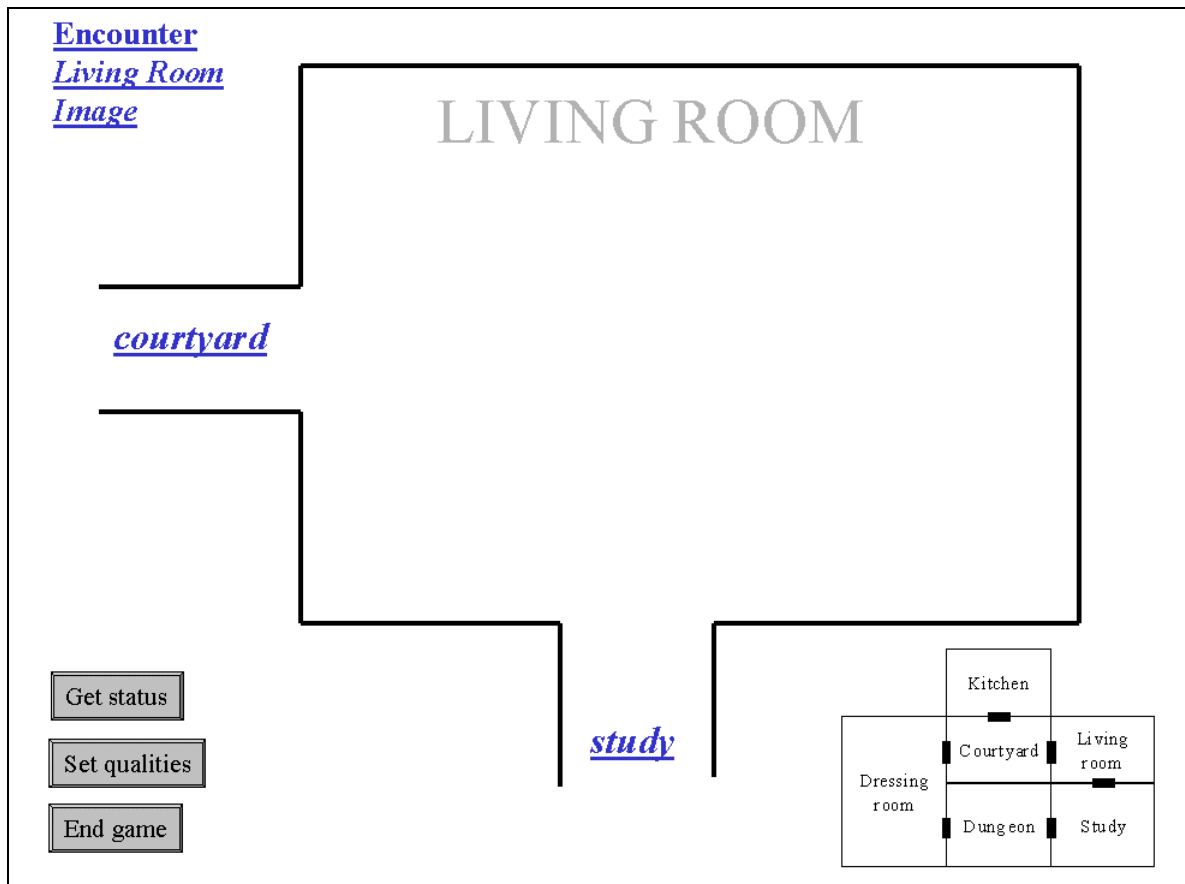


Figure 4.63 *Encounter Living Room Image*

### 3.2.AR.2.6 *Study area*

[essential; implemented] There shall be an area with name "study", requiring quality *concentration*. Its image is shown in [figure 4.64](#).

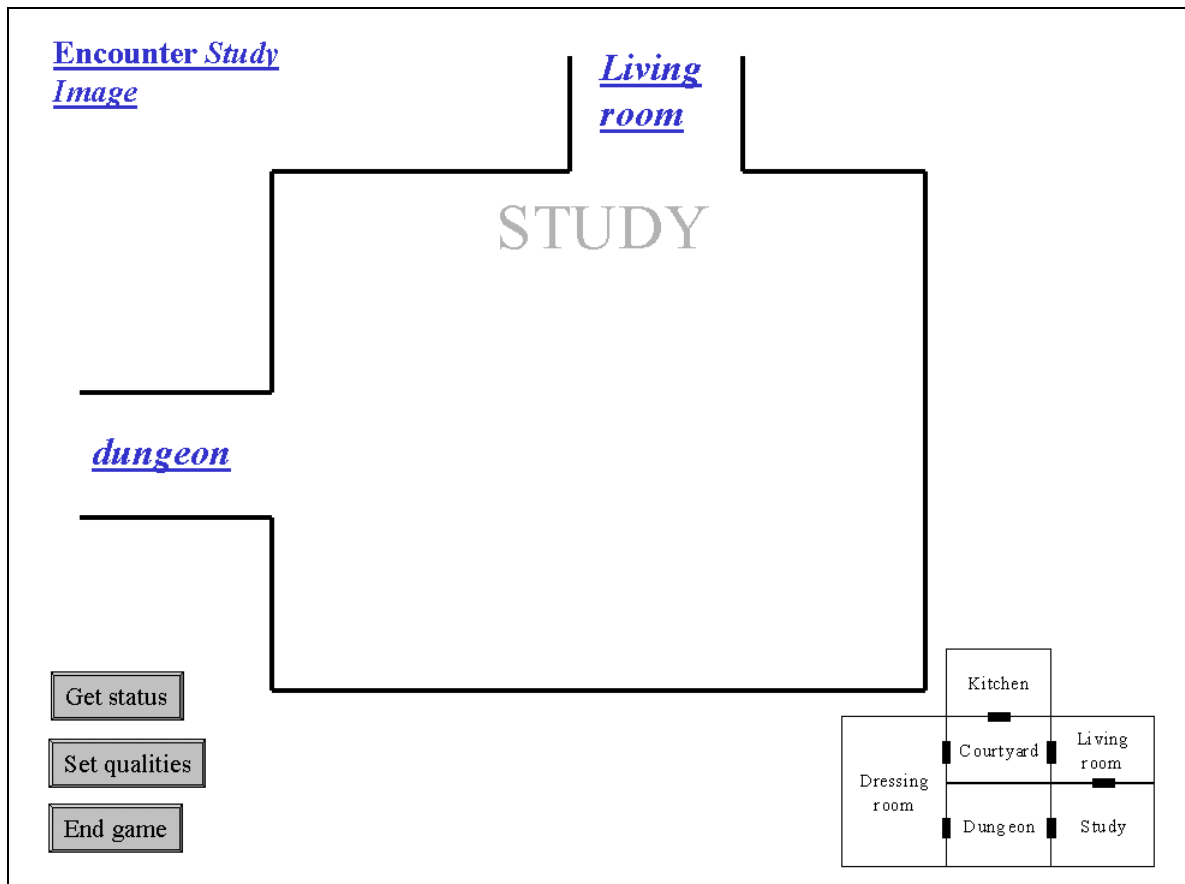


Figure 4.64 Encounter Study Image

### 3.2.AR.3 Area functionality

[Note to the student: this is the required functionality that pertains specifically to areas. Every functional capability of the application should belong to one of these sections.]

none

### 3.2.AR.4 Events pertaining to areas

[Note to the student: we separate the *events* that pertain to areas from the attributes, objects, and methods. An event is an action that occurs to the application, and which is instigated from outside of it.]

#### 3.2.AR.4.1 Display on entry of player character

[essential; implemented] Whenever the player's main character enters an area, that area and the characters in it shall be displayed on the monitor, filling the monitor.

#### 3.2.AR.4.2 Handling engagements

[essential; implemented] When a foreign game character enters an area containing the player's main character, or vice versa, they engage each other.

Note: Contents of Revision 1 sections 3.2.AR.4.3-6 either deleted or moved.

### ***3.2.AR.4.7 Display on entry of foreign character***

[essential; implemented] Whenever the foreign character enters the area in which the player is present, that area and the characters in it shall be displayed.

## **3.2.CH Connection hyperlinks between areas**

Connection hyperlinks are hyperlinks placed at each area exit, showing the area to which it is connected.

### **3.2.CH.1 Attributes of connections hyperlinks**

#### ***3.2.CH.1.1 Connection***

[essential; implemented] Each connection hyperlink corresponds to an area connection.

#### **3.2.CH.2 Connection hyperlink entities**

[essential; implemented] There are two connection hyperlinks corresponding to each area connection, one in each area of the connection.

#### **3.2.CH.3 Functionality of connection hyperlinks**

none

#### **3.2.CH.4 Events pertaining to connection hyperlinks**

##### ***3.2.CH.4.1 User clicks on a connection hyperlink***

The effect of clicking a connection hyperlink is that the player's character is displayed in the area on the other side of the area connection.

## **3.2.CO Connections between areas**

Characters travel from area to adjacent area by means of connections. Each of these connects two areas. [Figure 4.65](#) shows the required connections among the areas.

## Encounter Area Configuration (Desirable Requirement)

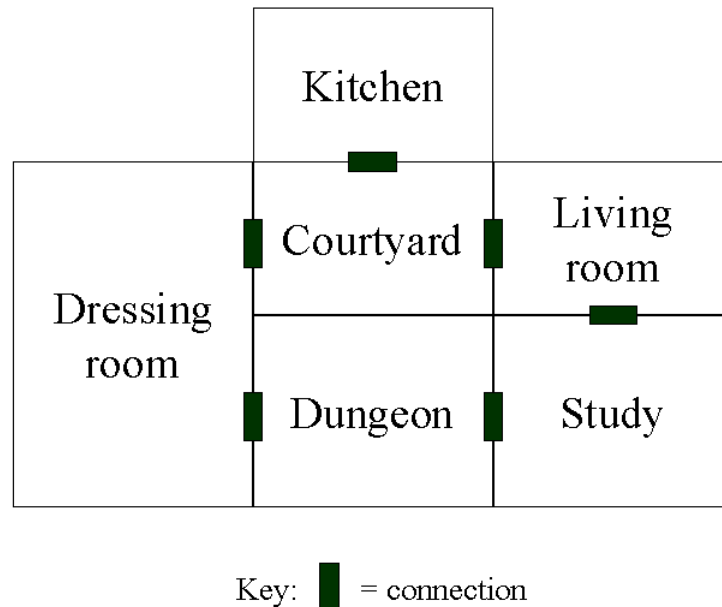


Figure 4.65 *Encounter Area Configuration (Desirable Requirement)*

### ***3.2.CO.1 Attributes of connections between areas***

#### ***3.2.CO.1.1 First and second areas***

[essential; implemented] Each connection will connect a pair of areas, which we will call the "first" and "second".

### **3.2.CO.2 Connections entities**

#### ***3.2.CO.2.1 Dressing room - courtyard***

[essential; implemented] There will be a connection between the dressing room and the courtyard.

#### ***3.2.CO.2.2 Dungeon - study***

[essential; implemented] There will be a connection between the dungeon and the courtyard.

#### ***3.2.CO.2.3 Study - living room***

[essential; implemented] There will be a connection between the dungeon and the living room.

### ***3.2.CO.2.4 Courtyard - living room***

[essential; implemented] There will be a connection between the living room and the courtyard.

### ***3.2.CO.2.5 Dressing room - dungeon***

[essential; implemented] There will be a connection between the dressing room and the dungeon.

### ***3.2.CO.2.6 Courtyard - kitchen***

[essential; implemented] There will be a connection between the kitchen and the courtyard.

## **3.2.CO.3 Functionality of area connections**

none

## **3.2.CO.4 Events pertaining to area connections**

### ***3.2.CO.4.1 Moving a character through a connection***

[essential; implemented] Connections are displayed as hyperlinks at the borders of areas whenever the player's character is in the area. When the user clicks such a hyperlink, the linked area is displayed, with the character in this area.

[*Query: This seems to duplicate 3.2.CH.4.1 – Discuss/Check*]

## **3.2.EC Encounter characters**

### **3.2.EC.1 Attributes of Encounter characters**

#### ***3.2.EC.1.1 Name of Encounter characters***

[essential; implemented] Every game character in the *Encounter* video game shall have a unique name. The specifications for names shall be the same as those for Area names, specified in 3.2.AR.1

#### ***3.2.EC.1.2 Qualities of Encounter characters***

[essential; implemented] Every game character has the same set of qualities. Each quality shall be a non-negative floating point number with at least one decimal of precision. These are all initialized equally so that the sum of their values is 100. The value of a quality cannot be both greater than zero and less than 0.5.

For the first release the qualities shall be *concentration*, *intelligence*, *patience*, *stamina*, and *strength*.

### **3.2.EC.1.3 Image of Encounter characters**

[essential; implemented] Every game character shall have an image.

### **3.2.EC.2 Encounter character entities**

The characters of the game are described among the types of *Encounter* characters.

### **3.2.EC.3 Functionality of Encounter characters**

#### **3.2.EC.3.1 Living points**

[essential; implemented] The *Encounter* game shall be able to produce the sum of the values of any character's qualities, called its *living points*.

#### **3.2.EC.3.2 Configurability of Encounter character quality values**

[essential; implemented] Whenever an *Encounter* character is alone in an area, the value of any of its qualities may be set. The value chosen must be less than or equal to the sum of the quality values. The values of the remaining qualities are automatically adjusted so as to maintain their mutual proportions, except for resulting quantities less than one, which are replaced by zero quality values.

### **3.2.ED Engagement displays**

[essential; implemented] There shall be a window displaying the result of engagements. The format is shown in [figure 4.66](#).

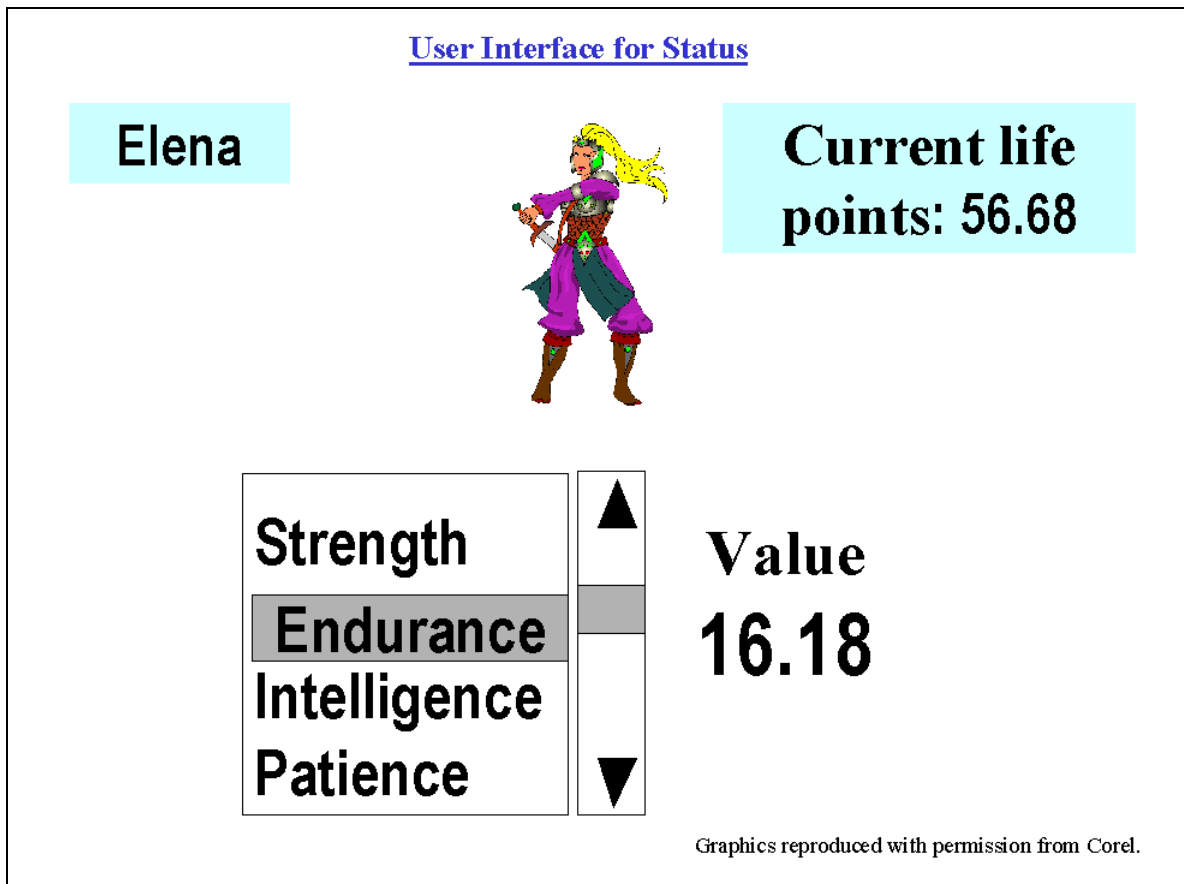


Figure 4.66 User Interface for Showing Status

### 3.2.ED.4 Engagement display events

#### 3.2.ED.4.1 *Dismissing the display*

[essential; implemented] When the user hits OK, the display disappears.

### 3.2.EG The *Encounter* game

The requirements in this section pertain to the game as a whole.

#### 3.2.EG.1 Attributes of the *Encounter* game

##### 3.2.EG.1.1 *Duration*

[optional; implemented] A record will be kept of the duration of each game, timed from when the player begins the game.

##### 3.2.EG.1.2 *Action buttons*

[Note to the student: this requirement was modified and moved from "AR".]

Every game window shall show a "Get status" button, a "Set qualities" button and an "End game" button in the lower left corner.

## **3.2.EG.2 Entities of the *Encounter* game**

### **3.2.EG.2.1 *Single game***

[essential; implemented] There shall be a single game.

[*Note to the student:* future releases will allow several versions of the game to run at the same time.]

## **3.2.EG.4 Events the *Encounter* game**

[*Note to the student:* transferred from AR.]

### **3.2.EG.4.4 *Pressing the Set qualities button***

When the user presses the *set qualities* button, and provided that there is no foreign character in the area, a window for setting the values of qualities appears, superimposed on the area. See 3.2.PQ for the specifications of this window.

### **3.2. EG.4.5 *Pressing the End game button***

[optional; not yet implemented] When the user presses the *end game* button, the game terminates. No additional screens appear.

[*Note to the student:* the last sentence, and inverse requirement, was felt necessary because games usually do display a summary of the session.]

### **3.2. EG.4.6 *Pressing the Get status button***

[optional; not yet implemented] When the user presses the *get status* button, an engagement display window appears showing the status of the player's character before and after the last engagement.

## **3.2.EN Engagements**

An *engagement* is the interaction between a game character controlled by the player and a foreign character.

### **3.2.EN.1 Attributes of engagements**

none

### **3.2.EN.2 Engagement entities**

There are no permanent engagement entities.

### 3.2.EN.3 Functionality of engagements

#### 3.2.EN.3.1 Engaging a foreign character

[Note to the student: this particular requirements is mathematical in nature, and so there is no attempt to replace the mathematics with natural language, at the risk of compromising its precision. The use of natural language to explain the mathematics is a good practice, however.]

[essential; implemented]

When an engagement takes place, the "stronger" of the two characters, is the one whose values of area-specific qualities sum to the greater amount. The system transfers to the stronger, half the values of each area-specific quality of the weaker. No transfer of points takes place if no character is stronger.

After the value re-allocations are made, if either character has no points, the game ends. If the game does not end, the player's character is moved to a random area, and the results of the engagement are displayed.

As an example of the value re-allocations, suppose that the player engages a foreign character in an area preferring *stamina* and *concentration*. If  $p_s$  is the value of the player's stamina etc., and assuming  $p_s + p_c > f_s + f_c$ , we would have  $p_s' = p_s + f_s/2$ ,  $p_c' = p_c + f_c/2$ ,  $f_s' = f_s/2$ , and  $f_c' = f_c/2$  where  $x'$  is the value of  $x$  after the transaction.

[The reader will recognize the defect in the last equation, which should be  $f_c' = f_c/2$ .

We will leave the defect intact as an example.]

To take a numerical example of an engagement in this area: if the player's stamina value is 7, and concentration value is 19, and Freddie the foreigner's stamina is 11 and concentration 0.6, then the player is stronger. The result of the engagement would be:

Player: stamina  $7 + 11/2 = 12.5$ ; concentration  $19 + (0.6)/2 = 19.3$

Freddie: stamina  $11/2 = 5.5$ ; concentration 0 because  $(0.6)/2$  is less than 0.5

### 3.2.EN.4 Events on engagements

#### 3.2.EN.4.1 Interrupting engagements

[implemented] Players are able to interrupt engagements on a random basis. On average, the player can stop one of every ten engagements, by executing the procedure to set qualities. The user tries to interrupt an engagement by attempting to set the player's qualities. If the game does not allow this, no indication is given: the game proceeds as if the attempt had not been made.

### 3.2.FC Foreign characters

A foreign character is an *Encounter* character not under the player's control.

#### 3.2.FC.1 Attributes of foreign characters

See *Encounter* character requirements. These are initialized to be equal.

[In future releases, foreign characters may mutate into new forms.]

#### 3.2.FC.2 Foreign character entities

[*Note to the student:* this section tells that there is only one foreign character.]

##### 3.2.FC.2.1 Freddie foreign character

[essential; implemented] There shall be a foreign character named "Freddie", whose image is shown in [figure 4.67](#). This character shall initially have a total of 100 points, distributed equally among its qualities.

#### Foreign Character Freddie's Image



Graphics reproduced with permission from Corel.

Figure 4.67 Image for Freddie Foreign Character

### 3.2.FC.3 Functionality of foreign characters

#### 3.2.FC.3.1 Foreign character movement

[essential; implemented] As long as it is alive, a foreign character should move from area to adjacent area at random intervals averaging two seconds. After being present in an area for a random amount of time averaging one second, all of the player's life points are divided among the qualities relevant to the area, such that the value of these qualities are as close to equal as possible.

### 3.2.PC Player characters

These are *Encounter* characters under the control of the player.

#### 3.2.PC.1 Attributes of player characters

See *Encounter* character attributes. Player character images can be selected from one of the images in [figure 4.68](#).

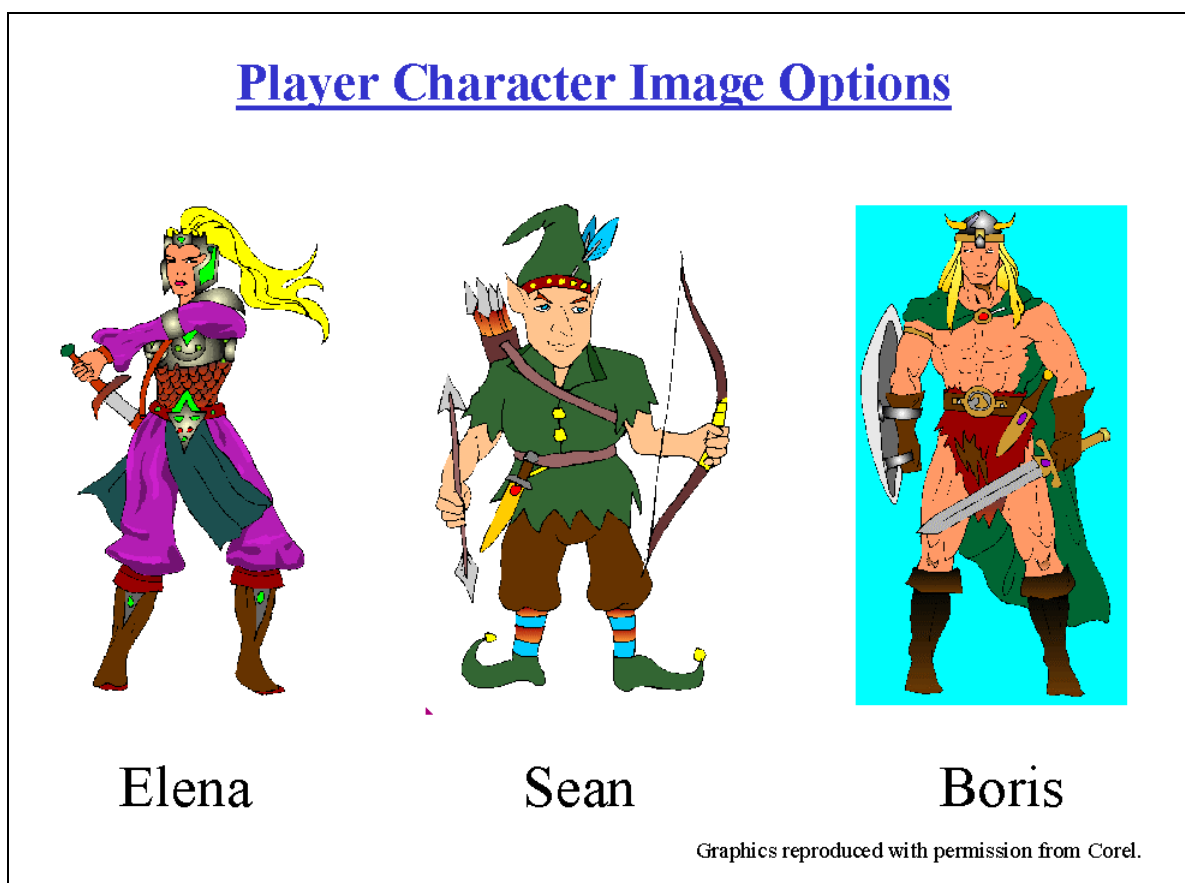


Figure 4.68 Player character image options

## **3.2.PC.2 Player character entities**

### ***3.2.PC.2.1 Player's main character***

The player shall have control over a particular game character called the "main" character. The nature of this control is subject to the restrictions specified in the remaining requirements. This character shall initially have a total of 100 points, distributed equally among its qualities.

### ***3.2.PC.2.2 Additional characters under the control of the player***

[optional; not yet implemented] The player shall be able to introduce characters he controls other than the main player. Details are to be decided.

## **3.2.PC.3 Player character functionality**

### ***3.2.PC.3.1 Configurability of the player character quality values***

[essential; implemented] Whenever all foreign players are absent from the area containing the player's main character, the player may set the value of any quality of the character using the *PlayerQualityWindow* shown in figure [4.69](#). The value chosen must be less than or equal to the sum of the quality values. The values of the remaining qualities are automatically adjusted so as to maintain their mutual proportions, except for resulting quantities less than 0.5, which are replaced by zero quality values.

### ***3.2.PC.3.2 Configurability of the player character images***

[desirable; not yet implemented] The player shall have the option to choose the image representing his or her main character from at least two images. These options are shown in the figure.

### ***3.2.PC.3.3 Aging of the player character images***

[optional; not yet implemented] ("Player aging") The main player character shall automatically increase each quality by a percentage for the first half of his or her life, then decrease them by the same percentage for the second half. Details are to be decided.

## **3.2.PQ The player quality window**

This is a window from which the player may allocate the values of his or her characters.

### 3.2.PQ.1 Attributes of the player quality window

The window for setting the qualities of a player character in *Encounter* is shown by means of a typical example in [figure 4.69](#).

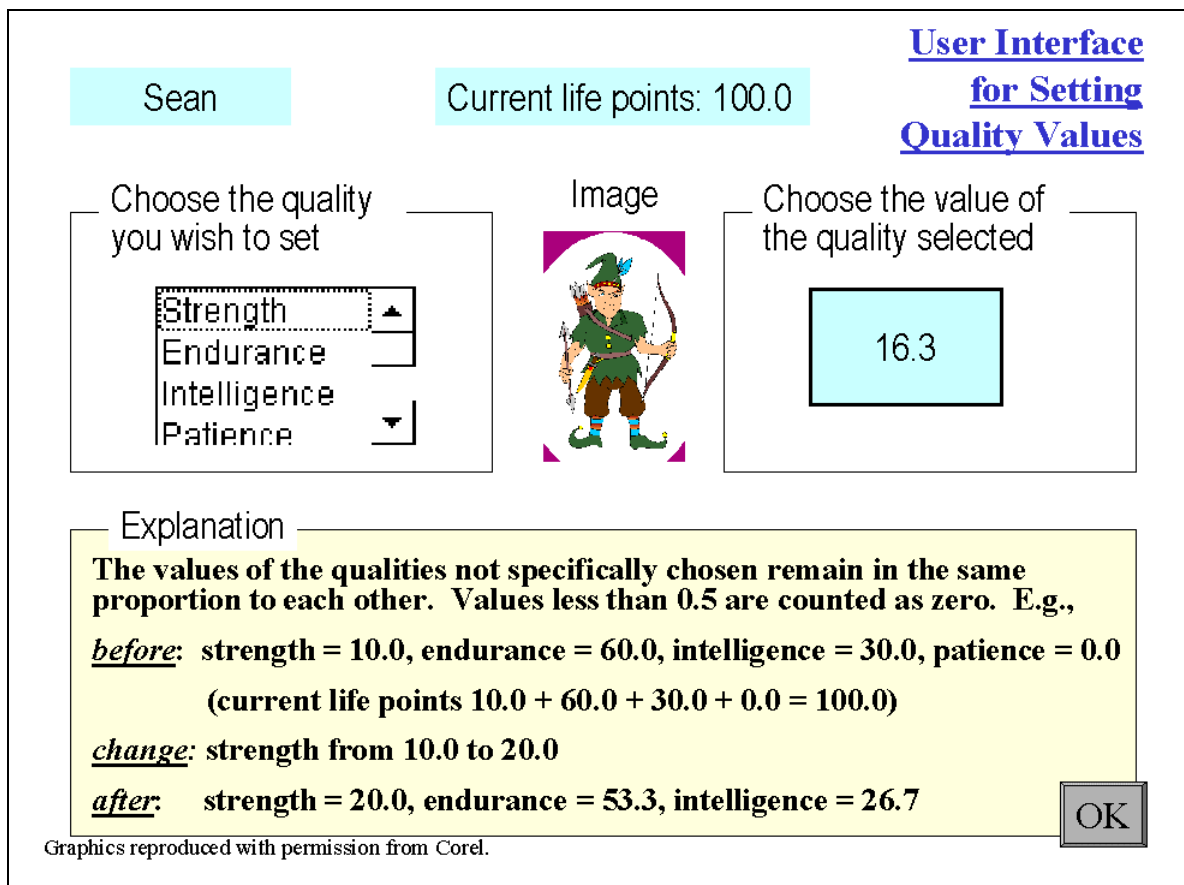


Figure 4.69 User Interface for Setting Values

The game character icon appears in the center and its name at the left top. The character's life points appear in the right top corner. On the left center is a list box of the qualities displaying four of them at a time. Clicking on one of these allows the player to select a value for it in the text box on the right. An explanation of how the arithmetic is performed is shown in a pale yellow box at lower center. Color backgrounds for the name, life points and value boxes are to be pale turquoise.

The method for performing quality value computations is described in section 3.2 below.

### 3.2.PQ.2 Player quality window entity

#### 3.2.PQ.2.1 Window for allocating qualities

[essential; implemented] A window shall be available under the conditions described above to allocate the values of the player character. The window shall have the appearance of the GUI shown in section 3.1.1.2 of this specification.

### **3.2.PQ.3 Player quality functionality**

#### ***3.2.PQ.3.1 Initiating the display***

[essential; implemented] The player quality menu shall be able to display itself.

### **3.2.PQ.4 Player quality window events**

#### ***3.2.PQ.4.1 Displaying the value of a quality***

[essential; implemented] When the player clicks on a quality in the list box on the left, the value of that quality shall be displayed in the text box on the right.

#### ***3.2.PQ.4.2 Setting the value of a quality***

[essential; implemented] When the user enters a legitimate value for a quality, and hits the "enter" button, the value of that quality **may be** set to the amount entered, **subject to requirement 3.2.PQ.4.3**. If the value is invalid, an error **message** shall appear.

#### ***3.2.PQ.4.3 Dismissing the window***

[essential; ~~not yet~~ implemented] When the user hits the OK button, a time of four seconds elapses, after which the window disappears. At the end of this time period (i.e., if there are no interruptions), the value allocations made.

#### ***3.2.PQ.4.4 Interruption***

[essential; implemented] On interruption of the quality value window being displayed, the window vanishes.

Note that interruptions will be caused by a foreign character entering the area. Note also that the quality values are not changed, and an engagement takes place.

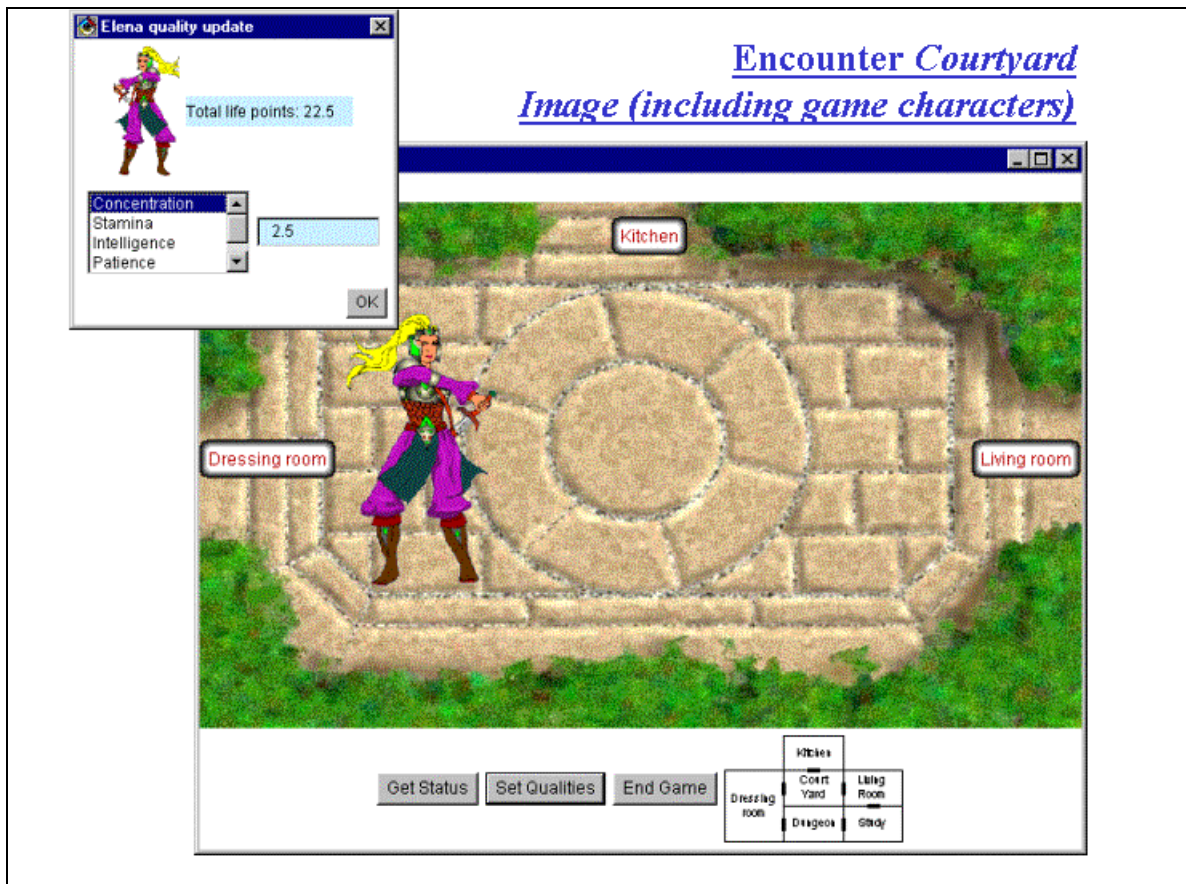
### **3.2.TH Thumbnail sketch of playing areas**

This is a figure showing the relationship among the areas in the immediate environment of the player.

#### **3.2.TH.1 Attributes of the thumbnail sketch**

##### ***3.2.TH.1.1 Thumbnail image***

[essential] The thumbnail image shall be as [shown below](#)(?):



### 3.3 Performance requirements

[Note to the student: these include required speeds and/or time to complete. Unless documented in a different section of the SRS, they may also include memory usage (RAM and/or disk), either statically, or dynamically (i.e., memory required at runtime).]

The application shall load and display the initial image in less than a minute.

Engagements should execute in less than one second.

These requirements are tested in STD < reference to test goes here >.

### 3.4 Design constraints

[Note: this section specifies restrictions on the design. If there is no material in this section, designers are free to create any (good) design that satisfies the requirements. To take analogy, we can add the design requirement "one-story" to the following. "A house with 4 bedrooms, all of which are less than a thirty-second walk from the family room." ]

*Encounter* shall be designed using UML and object-oriented design. It shall be implemented in Java. The software will run as a Java application on Windows 95. It shall be designed in a way that makes it relatively easy to change the rules under which the game operates so that others can customize the game.

## 3.5 Software system attributes

### 3.5.1 Reliability

*Encounter* shall fail not more than once in 1000 encounters. Test documentation < reference to test goes here>.

### 3.5.2 Availability

*Encounter* shall be available for play on any PC running Windows 95 only (i.e., no other applications simultaneously). Test documentation < reference to test goes here>.

### 3.5.3 Security

[future releases will allow access to saved games only with a password]

### 3.5.4 Maintainability

#### 3.5.4.1 Changing characters and areas

[essential] It shall be relatively straightforward to change characters and areas.

#### 3.5.4.2 Globally altering styles

[desirable] It shall be straightforward to globally alter the style of the areas and connections. (Style changes reflect different levels of game play in the same environment.)

#### 3.5.4.3 Altering rules of engagement

[optional] rules of engagement should be relatively easy to change

## 3.6 Other requirements

none

## 4. Supporting information

none

### 4.1 Table of contents and index

To be supplied (See first pages in fact)

### 4.2 Appendixes

To be supplied

[*Note to the student:*

Appendices may include

- a. sample I/O formats, descriptions of cost analysis studies, or results of user surveys,
- b. supporting or background information that can help the readers of the SRS,

- c. a description of the problem to be solved by the software,
- d. special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements

State explicitly whether or not each appendix is to be part of the SRS.]